



TITLE:

データベース問合せ言語の変換アルゴリズムについて(アルゴリズムの数学的基礎理論とその応用)

AUTHOR(S):

宮尾, 淳一; 富永, 一幸; 菊野, 亨

CITATION:

宮尾, 淳一 ...[et al]. データベース問合せ言語の変換アルゴリズムについて(アルゴリズムの数学的基礎理論とその応用). 数理解析研究所講究録 1986, 591: 84-94

ISSUE DATE:

1986-05

URL:

<http://hdl.handle.net/2433/99475>

RIGHT:

データベース問合せ言語の変換アルゴリズムについて

On Optimization in Query Translation from AQL to SQL

広島大学 工学部 宮尾 淳一 (Jun'ichi MIYAO)

富永 一幸 (Kazuyuki TOMINAGA)

菊野 亨 (Tohru KIKUNO)

1. まえがき

筆者らは、現在、データベースシステムAIDE-II (An Intelligent Database System for End Users-II) を開発中である。このシステムでは従来の関係データベースとエンドユーザの間に意味データモデルを導入し、その意味データモデル上で問合せを記述する⁽³⁾。これにより、データベーススキームに関する知識を持たなくても問合せの作成ができる。

更に、筆者らが開発した表形式言語AQLを利用すれば、問合せを視覚的に作成することが可能となる⁽²⁾。言語AQLの特徴的な機能にユーザビューがある。これは機能的には関係データベースにおけるビューと同様であるが、ユーザビューの場合には予め定義しておく必要なく自由に指定できるという特長がある。そのため、エンドユーザでも容易にユーザビューを利用して問合せを記述することができる。この問合せは関係データベース上の問合せに変換された後、通常の問合せ処理の手順に従って実行される。ところが、この問合せの変換は一義的に定まらず、一般に、あいまいさを含む。本稿では、このあいまいさを実用性の立場から解消する試みについて議論する。

2. ユーザビュー

問合せ言語 AQL⁽²⁾は QBE⁽⁵⁾と同じ表形式言語であり，そのシンタックスは QBE と同様である．関係データベースに対する一般的な機能である問合せ作成，ビュー定義，データ定義⁽⁴⁾などは AQL にも含まれている．AQL では，エンドユーザでも容易に問合せを作成できる機能としてユーザビューを新たに提供している．

[例 1] 関係データベース内に存在する次の 3 つの関係について考える．

MEMBERS(#MEMBER, NAME, ADDRESS)

ORDERS1(#ORDER1, #MEMBER1, ITEM1, QUANTITY1)

ORDERS2(#ORDER2, #MEMBER2, ITEM2, QUANTITY2)

意味データモデル上で値域 number, name, address, item が定義され，値域と属性との対応も与えられているとする．ここでは，特に，値域 item に 2 つの属性 ITEM1, ITEM2 が対応すると仮定する．今，次の問合せ Q_1 を考える．

Q_1 = "PC-9801 を注文した人の名前と住所を求めよ"．

この Q_1 は AQL のユーザビューを用いると，図 1 のように記述できる．ユーザビューでは表の左側に必要な値域の名前を列挙し，右側に値域が満たすべき条件を記述する．

	USER VIEW
name	
address	
item	PC-9801

図 1 ユーザビューによる問合せ Q_1 の記述

問合せ Q_1 に対応する関係データベースの問合せの例を図2の Q_{11} , Q_{21} に示す. 同図で, " " は変数を表し, "P." は出力すべき属性を指定している. 与えられたユーザビュー Q_1 に基づいて問合せ Q_{11} , Q_{21} を構成する手順の詳細については3.で述べる. ここではその基本方針について直観的な説明を行う.

(A) 先ず, ユーザビュー中の各値域名に対応する属性名を求める. 同時に, 関係データベース内に存在するその属性名を含む関係名の集合 R_1, R_2, \dots を求める.

(B) 次に, 各 R_i に対し, R_i 中のすべての関係を結合する問合せ Q_i を生成し, 各 Q_i の結合結果の和をとる. なお, 余りに多くの結合を必要とする Q_i は予め削除する.

この例の場合には, ステップ(A) によって,

$$R_1 = \{ \text{MEMBERS, ORDERS1} \}, R_2 = \{ \text{MEMBERS, ORDERS2} \}$$

が得られる. 次に, ステップ(B) によって, 図2に示す Q_{11} , Q_{21} が構成される. □

MEMBERS	RETRIEVE
#MEMBER	<u> </u> X
NAME	P.
ADDRESS	P.

ORDERS1	RETRIEVE
#ORDER1	
#MEMBER1	<u> </u> X
ITEM1	PC-9801
QUANTITY	

Q_{11}

MEMBERS	RETRIEVE
#MEMBER	<u> </u> X
NAME	P.
ADDRESS	P.

ORDERS2	RETRIEVE
#ORDER2	
#MEMBER2	<u> </u> X
ITEM2	PC-9801
QUANTITY	

Q_{21}

図2 関係データベースの問合せ Q_{11} と Q_{21}

3. 変換手順

ここでは、ユーザビューで記述された問合せに対する変換手順の概要を述べる。但し、2.で説明した基本方針のステップ(B)についてのみ述べる。対象となる関係データベース \mathcal{R} と各関係 R_i を構成する属性名の集合は与えられているとする。

[定義1] 関係データベース \mathcal{R} に対する結合グラフを無向グラフ $G = (V, E)$ と定める。ここで、 $V = \mathcal{R}$ とし、 $E = \{(R_i, R_j) \mid \text{2つの関係 } R_i, R_j (\in \mathcal{R}) \text{ は、共に、ある属性 } A \text{ を含んでいる}\}$ とする。このとき、各無向枝 $(R_i, R_j) \in E$ にはラベル A を付ける。以降では、このラベルを $\ell(R_i, R_j)$ で参照する。 \square

関係データベース上の結合操作のみを用いた問合せは結合グラフ G の部分グラフとして表現できる。この性質を利用した変換の概要を次に示す。なお、与えられるユーザビューに現れる(値域名に対応する)属性名をすべて含む関係の集合を V_0 で表し、ユーザビューで必要とされる関係の集合と呼ぶ。この集合 V_0 は2.のステップ(A)を適用して既に求まっていると仮定する。

ステップ1：結合グラフ G 上の連結な部分木であって、しかも、集合 V_0 の要素のすべてを節点として含む木を列挙する。今、列挙された木の集合を \mathcal{T} で表す。

ステップ2： \mathcal{T} 中に含まれる木の枝が結合操作に対応しているの
で、 \mathcal{T} に対し共通な枝を1つにまとめるという最適化を行う(問題JM)。問題JMの解として、結合処理に必要なページフェッチの回数が最小となる集合 \mathcal{T}' を \mathcal{T} から構成する。

ステップ3： \mathcal{T}' 中の木の各枝に対応する結合処理を実行し、それらの集合和をとって最終的に問合せの結果を求める。

筆者らは、既に、ステップ1を実用的時間内で解くアルゴリズムJTを提案している。シミュレーション実験によって、実用的環境下ではステップ1が1.5秒以内で解けることを確認している⁽²⁾。使用したコンピュータは日本・データゼネラル社のECLIPSE MV/4000である。

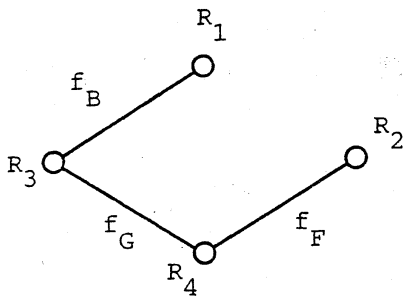
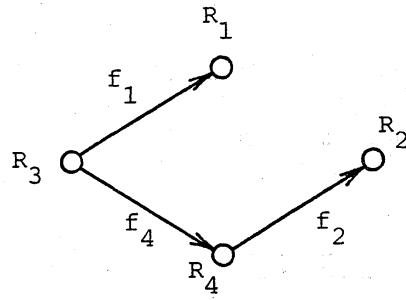
4. 結合回数の最小化

関係データベースにおける結合は最も時間のかかる基本操作である。従って、問合せの処理に当たっては結合回数の減少を図ることが重要となる。但し、共通な結合操作を1度実行しておいてその結果を複数回参照する場合、必要なディスクのページフェッチの総数が増大する危険性がある。以下では、文献(1)に従って各結合木の処理に必要なページフェッチの数を見積り、それに基づいて最適化問題を定式化する。

4.1 処理コスト

変換のステップ1の出力である各結合木 $T = (V_T, E_T)$ に対して、(1)各節点 $R_i \in V_T$ が表す関係中に含まれるタプル数 N_i とその格納に必要なディスクのページ数 M_i 、(2)各無向枝 $(R_i, R_j) \in E_T$ 、 $\ell(R_i, R_j) = \alpha$ に対して、 R_i と R_j に対する選択率 f_α (2つの関係 R_i, R_j の結合結果に含まれるタプル数の割合)、が与えられているとする。

次に、結合木 $T = (V_T, E_T)$ と T 上の任意の節点 $r \in V_T$ に対し、 r を根とし、 T 上の各枝に根から葉へ方向を与えて求まる有向木を $TR_r(T) = (V'_T, E'_T)$ と表す(図3参照)。なお、各有向枝 $(R'_i, R'_j) \in E'_T$ に対しては、対応する無向枝 $(R_i, R_j) \in E_T$ と同じ値の選択率を与える。従って、図3において $f_1 = f_B$ 、 $f_2 = f_F$ 、 $f_4 = f_G$ となる。

(a) $T = (V_T, E_T)$ (b) $TR_{R3}(T) = (V'_T, E'_T)$ 図3 結合木 T と有向木 $TR_{R3}(T)$

ここでは，各結合木に対する実行形式として文献(1)と同様，入れ子ループ法 (nested loops method) を仮定する．今， $|V'_T| = n + 1$ とする．入れ子ループ法における一番外側のループを R_{i0} ，一番内側のループを R_{in} と表す．文献(1)より，有向木 $TR_r(T)$ と系列 $S = (R_{i1} R_{i2} \cdots R_{in})$ が与えられたとき， $TR_r(T)$ の S に関する処理に必要なコスト $C(S)$ は次のように再帰的に定義される⁽¹⁾．但し，系列 S に対し，各 $R_{ij} \in V'_T$ であり，かつ S 中の各節点 R_{ij} ($1 \leq j \leq n$) の現われる節点の順序は，有向木 $TR_r(T)$ 上の枝の向きによって定まる節点上の半順序関係を満たすものとする．

(i) $C(\Lambda) = 0$ (Λ は空系列)

(ii) $C(R_i) = f_i M_i$

(iii) $C(S_1 S_2) = C(S_1) + T(S_1) C(S_2)$

但し， $T(S)$ は次のように定義される．

(i) $T(\Lambda) = 1$ (Λ は空系列)

(ii) $T(S) = f_{i1} N_{i1} f_{i2} N_{i2} \cdots f_{ik} N_{ik}$

(但し， $S = R_{i1} R_{i2} \cdots R_{ik}$)

1つの結合木 $T = (V_T, E_T)$ が単独に与えられたとき，コスト $C(S)$ が最小となる最適な結合順序 S は文献(1)の Algorithm B によって求めることができる．この最適な結合順序を \mathfrak{S}_T と表すと，結合木

の集合 $\mathcal{T} = \{T_1, T_2, \dots, T_n\}$ の処理コストは, $\text{COST}(\mathcal{T}) = \sum_{T \in \mathcal{T}} C(\mathcal{S}_T)$ と定義される.

4.2 問題 JM

先ず, 結合木の集合 \mathcal{T} 中に含まれる共通な結合を 1 つにまとめる変換 CJ を定義する.

[定義 2] 結合木の集合 \mathcal{T} と木 $T_k \in \mathcal{T}$ 上の無向枝 (R_i, R_j) ($\ell(R_i, R_j) = \alpha$) に注目する. ラベル, 及び, 端点の節点名を含めて同じ枝 (R_i, R_j) が $T_{k\ell} (\in \mathcal{T})$ ($\ell = 1, 2, \dots$) にも含まれているとする. このとき, $T_k, T_{k\ell} (\ell = 1, 2, \dots)$ 上でそれぞれ枝 (R_i, R_j) を短絡除去し, 新しい結合木 $T_{n+1} = (\{R_i, R_j\}, \{(R_i, R_j)\})$, $\ell(R_i, R_j) = \alpha$ を \mathcal{T} に追加する操作を, ここでは枝 (R_i, R_j) に対する変換 CJ と定める. □

以上の準備の下に, 結合回数の最小化問題 JM を次のように定式化する.

[問題 JM] ステップ 1 で求まった結合木の集合 \mathcal{T} に対して, 変換 CJ を有限回適用して, $\text{COST}(\mathcal{T}')$ が最小となる結合木の集合 \mathcal{T}' を求めよ. □

[例 2] 図 4 に示す $\mathcal{T} = \{T_1, T_2\}$ を考える. 今各 T_i ($i=1, 2$) に対するパラメータとして $N_1=1024, N_2=512, N_3=2048, N_4=4096, M_1=32, M_2=16, M_3=64, M_4=128$, 選択率 $f_A=0.00025, f_B=0.01, f_C=0.005, f_D=0.0025$ を考える. このとき, T_1 と T_2 上に存在する枝 (R_2, R_3) ,

(R_1, R_4)に変換C Jが適用可能である．先ず，(R_2, R_3)に変換C Jを適用すると集合 $\mathcal{T}' = \{T_1, T_2, T_3\}$ が求まる．引き続き(R_1, R_4)に適用すると集合 $\mathcal{T}'' = \{T_1, T_2, T_3, T_4\}$ が求まる（図5参照）．このとき， $\mathcal{T}, \mathcal{T}', \mathcal{T}''$ の処理コストはそれぞれ次のようになっている．

$$\text{COST}(\mathcal{T}) = 2.48 \times 10^8, \text{COST}(\mathcal{T}') = 1.74 \times 10^8, \text{COST}(\mathcal{T}'') = 3.54 \times 10^9$$

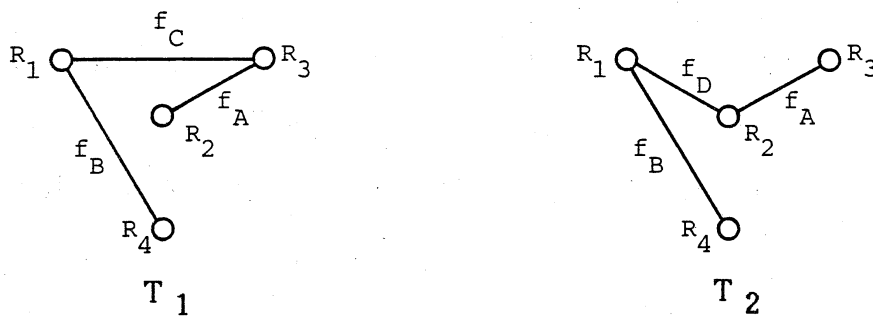


図4 結合木の集合 \mathcal{T}

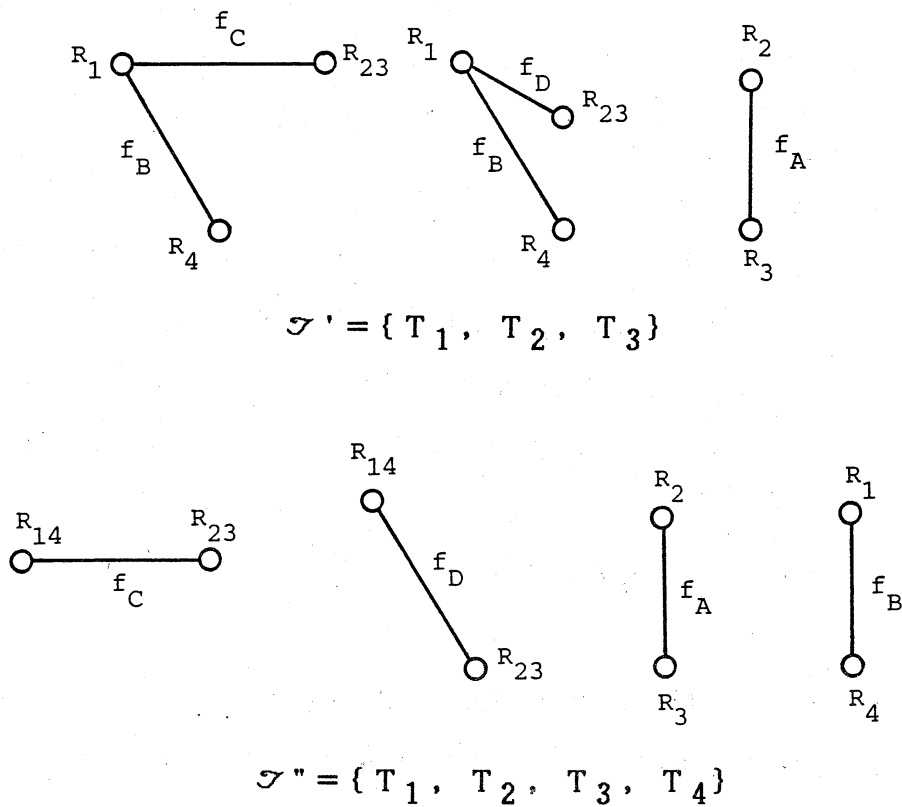


図5 結合木の集合 \mathcal{T}' と \mathcal{T}''

結合回数だけ(すなわち, 枝の総数だけ)に注目してみると \mathcal{T} が最小であるが, ディスクのページフェッチの数まで考慮した場合は \mathcal{T}' が最小となる. このように, 結合回数の最小化だけでは必ずしも実際の意味での最適化にまでは至らないことが分かる. \square

4.3 ヒューリスティック解法

次に, 問題 JM に対するヒューリスティックアルゴリズム JM を示す. アルゴリズム中で, $|(\mathcal{R}_i, \mathcal{R}_j)|, \ell(\mathcal{R}_i, \mathcal{R}_j) = \alpha$ は, 結合木の集合 \mathcal{T} 上で考えてラベル α の枝 $(\mathcal{R}_i, \mathcal{R}_j)$ を含む木の総数を表す.

[アルゴリズム JM]

ステップ 1 : $\mathcal{T}_{\min} := \mathcal{T}$, $C_{\min} := \text{COST}(\mathcal{T})$, $A := \cup \{E_T \mid T = (V_T, E_T) \in \mathcal{T}\}$ と初期化する.

ステップ 2 : A に属する枝の中から $|(\mathcal{R}_i, \mathcal{R}_j)|$ の値が最大の任意の枝を 1 つ選ぶ. いま, それを $(\mathcal{R}'_i, \mathcal{R}'_j)$ とすると, $A := A - \{(\mathcal{R}'_i, \mathcal{R}'_j)\}$ と更新する. もし, $|(\mathcal{R}'_i, \mathcal{R}'_j)| < 2$ ならば, 次はステップ 4 へ行く.

ステップ 3 : 枝 $(\mathcal{R}'_i, \mathcal{R}'_j)$ に変換 C J を適用して, その結果を \mathcal{T}' とおく. もし $\text{COST}(\mathcal{T}') < C_{\min}$ ならば, $\mathcal{T}_{\min} := \mathcal{T}'$, $C_{\min} := \text{COST}(\mathcal{T}')$, $\mathcal{T} := \mathcal{T}'$ と更新する. 次はステップ 2 へ行く.

ステップ 4 : \mathcal{T}_{\min} を出力する. \square

上述のアルゴリズム JM の性能評価を行うため, 日本・データゼネラル社の ECLIPSE MV/4000 上で C 言語を用いてプログラムを開発した. 各パラメータに対し一様乱数を用いてデータを生成し, シミュレーション実験を行った. 図 6 の ● にアルゴリズム JM で求めた COST を示

す。なお、同図の▲は(ページフェッチが結合回数に比例すると見なし
て結合回数の最小化を行った)文献(2)のヒューリスティックアルゴ
リズムで求めたCOSTを示している。この2つの結果を比較すると、結
合回数だけでなく、ディスクのページフェッチも定式化の段階で考慮
した方が問合せ処理コストのより一層の低減につながる事が分かる。

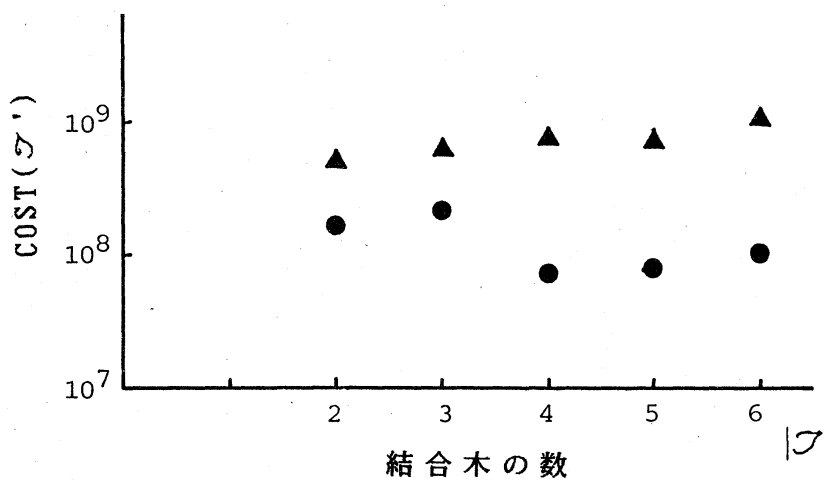


図6 シミュレーション実験

最後に、日頃ご指導頂いている広島大学工学部吉田典可教授、並び
に、文献(1)に関してご教示頂いた京都大学工学部茨木俊秀教授に深
謝します。

文献

- (1) Ibaraki, T. and Kameda, T.: "On the optimal nesting order for computing N-relational joins," ACM Trans. Database Syst., 9, 3, pp.182-502 (1984).
- (2) 宮尾, 富永, 菊野, 吉田: "表形式問合せ言語 AQL から SQL への変換について," 信学技報, AL 85-66, pp.69-79 (1986).

- (3) Sugihara, K., Miyao, J., Kikuno, T. and Yoshida, N.: "A semantic approach to usability in relational database systems," Proc. IEEE COMPDEC, pp.203-210 (1984).
- (4) Ullman, J.D.: "Principles of Database Systems," 2nd Ed., Computer Science Press (1982).
- (5) Zloof, M.M.: "QBE/OBE: A language for office and business automation," Computer, 14, 5, pp.13-22 (1981).